Computers Math. Applic. Vol. 24, No. 3, pp. 29–35, 1992
Printed in Great Britain.

0097-4943/92 $5.00 + 0.00

THE COMPUTER GENERATED SYMBOLIC APPROXIMATIONS TO SYSTEMS OF NONLINEAR ODE'S BY MATRIX ANNIHILATION AND THE NEWTON-KANTOROVICH METHOD

JAMES N. HANSON
Computer and Information Science Department
Cleveland State University
Cleveland, Ohio 44115, U.S.A.

(Received August 1990)

Abstract—The symbolic solution, $x = \exp(At)\,x(0)$, of the n-th order linear system, $dx/dt = Ax$, is approximated by using matrix annihilation of A, where the characteristic polynomial, but not its roots, is obtained by the methods of Leverrier or Krylov. This linear solution may be used to successively approximate the nonlinear case via Kantorovich's extention of Newton's method to Banach spaces. Symbolic solutions in the form of Taylor polynomials have been generated by the Formac extension to PL/I, to which very high order variable arithmetic has been added.

1. INTRODUCTION

A central problem in applied mathematics is solve the linear system of differential equations, $dx/dt = Ax$ and $x(0) = c$, explicity, in term of t, as opposed to representing the solution in terms of some transform variable. A is an $n \times n$ matrix. The solution to this system is $x(t) = \exp(At)\,c$. Hence, the problem reduces to evaluating the matrix exponential, $\exp(At)$. Much attention has been given to the expansion of the scalar case, $\exp(z)$, where z may be complex [1–4] and its extension to the matrix case [5–8]. Varga, in several papers, e.g., [9], has developed a particulary efficient expansion of $\exp(-x)$ in $[0, \infty)$ using rational Chebyshev approximations, which he applies to computing $\exp(A)$. Moler and Van Loan [10] provide an excellent review of methods for computing exponent function. The use of Padé approximations, or Chebyshev rational approximations, etc., are especially useful when fast and accurate expansions for fixed precision (i.e., hard code precision) are desired. However, such representations would be prohibitively complicated for the variable precision case, since they would have to be generated anew as the precision changes. This paper seeks to obtain symbolic solutions and where, essentially, no limit is placed on numeric precision. For this reason, solutions in Taylor polynomials have been used, since their coefficients can be easily and exactly obtained at execution time, and since polynomials are easily manipulated. In comparison, the rational Chebyshev representation of [9] possesses remarkable convergence properties but would require an enormous of time to generate dynamically for variable precision arithmetic. The Formac-interpreter extension of PL/I [11] has been used, providing 2300 decimal digit rational precision, and to which the author has added a very efficient arbitrary precision floating point arithmetic [12]. The basic solution, $\exp(At)\,c$, will be applied to the nonlinear case by employing Kantorovich's extension of Newton's method [13].

2. THE EXPANSION OF $\exp(AT)\,x_0$

Let $P(1, n : 0, n-1)$ be the partitioned matrix of column vectors,

$$P = (c, A\,c, A^2c, \ldots, A^{n-1}c),$$

whose row index ranges from 1 to n, and whose column index from 0 to $n-1$. Let m be the order of the polynomial solution for the x; where we have in mind that $m > n$. Define $B(n : m, 0 : n-1)$ to be the matrix whose first row contains the coefficients of the characteristic polynomial in increasing exponential order. The second row is obtained from the first row. The expansion of the characteristic polynomial, $|A - Iy| = 0$, provides an identity in the highest power, y^n, of y. Multiplying this identity through by y, and using the expression for y^n to eliminate y^n, yields an $n-1$ order polynomial for y^{n+1}. The coefficients of this resulting polynomial, in increasing exponential order, are the elements of the second row of B. Successive rows of B are obtained from the previous row in this manner. The data structures, P and B, provide an explicit expression of the x_i in terms of t, which is amenable to being developed by computer-performed symbol manipulation. The basic ingredient to this expansion is the Cayley-Hamilton matrix annihilation theorem as embodied in the B matrix. First, approximate $x(t)$ by the first $m+1$ terms of the MacLaurin expansion of $\exp(At)$,

$$x = \exp(At)\, c = \left(\sum_{j=0}^{m} A^j \frac{t^j}{j!} \right) x_0 = c + A\, c\, t + A^2\, c\, \frac{t^2}{2!} + \cdots + A^m\, c\, \frac{t^m}{m!}.$$

Let $P(*, j)$ be the cross sections of P, i.e., a column matrix defined to be the j-th column of P, then this last expression can be rewritten as

$$\begin{aligned} x = {} & P(*,0) + P(*,1)\, t + \cdots + P(*, n-1)\, \frac{t^{n-1}}{(n-1)!} \\ & + \left(B(n,0)\, P(*,0) + B(n,1)\, P(*,1) + \cdots + B(n, n-1)\, P(*, n-1) \right) \frac{t^n}{n!} \cdots \\ & + \left(B(m,0)\, P(*,0) + B(m,1)\, P(*,1) + \cdots + B(m, n-1) P(*, n-1) \right) \frac{t^m}{m!}. \end{aligned}$$

Next, factor on the $P(*, j)$, in order to obtain x as linear sum of the $P(*, j)$ with polynomials in t, $q(j)$, as coefficients,

$$x = \sum_{j=0}^{n-1} q(j)\, P(*, j),$$

or

$$x_i = \sum_{j=0}^{n-1} \left(\frac{t^j}{j!} + \sum_{k=n}^{m} B(k, j) \frac{t^k}{k!} \right) P(i, j).$$

A PL/I-Formac program, using the extension of [12], has been developed to construct and display the x_i [14]. The symbolic methods employed here attempt to avoid the problem of numeric round-off error. However, a truncation error is introduced by truncating the Taylor series. The resulting series may be generated and/or evaluated by rational arithmetic, the variable floating point arithmetic of [12] or by the 6, 15 or 33 decimal digits precision supported by the architecture of the IBM mainframe. Note that this method did not require the characteristic roots of A.

3. SYMBOLICALLY DEVELOPING THE CHARACTERISTIC POLYNOMIAL

The central problem in developing this expansion for x is symbolically constructing the characteristic polynomial of A. Leverrier's method [14] readily yields to symbolic manipulation. For floating point computations on a machine where division and multiplication are relatively slow operations, Krylov's method is clearly superior, requiring on the order of n^3 floating point multiplications-divisions, compared to n^4 for the Leverrier method. However, hardware floating point multiplications and divisions are not much lower than additions for a large mainframe,

such as the IBM 3081 used herein. Because of its simplicity, Leverrier's method is given. Leverrier's method generates the coefficients c_i, of the characteristic polynomial, from the intermediate matrices A_i and B_i, where

$$\begin{aligned} A_0 &= A, & c &= \mathrm{Tr}(A_0), & B_0 &= A_0 - c_0 I, \\ A_1 &= AB_0, & c_1 &= \frac{1}{2}\mathrm{Tr}(A_1), & B_1 &= A_1 - c_1 I, \\ &\cdots & &\cdots & &\cdots \\ A_{n-1} &= AB_{n-2}, & c_{n-1} &= \frac{\mathrm{Tr}(A_{n-1})}{(n-1)}. \end{aligned}$$

Tr (.) is the trace operator and c_j is the coefficient of y^j.

4. SAMPLE SOLUTIONS OF $dx/dt = Ax$

In order to test the algorithm matrices, A must be chosen for which the solution of $dx/dt = Ax$ is known. Random matrices were tested; however, they were only useful as time trials but not for verifying the algorithm. The following well known theorem, e.g., see Gantmacher [15], provides many test cases for which the powers of A are easily generated.

THEOREM. *Y and A are $n \times n$ matrices and f a non-constant polynomial. Let Y be diagonalizable, i.e., there exists an S such that $dg(y_1, \ldots, y_n) = S^{-1}YS$, and let $M = dg(z_1, \ldots, z_n)$, where $y_i = f(z_i)$. Then the solution of $f(A) = Y$ is $A = SMS^{-1}$.*

For example, this may be applied to the "square root" case [16, p. 28] where Y is the unit matrix and $f(u) = u^2$, to obtain a solution (i.e., a square root of the unit matrix),

$$(A)_{ij} = (-1)^{j-1}\,\mathrm{comb}(j-1, i-1),$$

where the combinatorial operator is taken to return zero if the first operand is less than the second. In this case, $A^2 = I$ and

$$\begin{aligned} x = \exp(At)\,c &= \left(I + At + A^2\frac{t^2}{2!} + \cdots\right)c \\ &= \left(I + At + I\frac{t^2}{2!} + A\frac{t^3}{3!} + \cdots\right)c = \left(\left(1 + \frac{t^2}{2!} + \cdots\right)I + \left(t + \frac{t^3}{3!} + \cdots\right)A\right)c \\ &= ((\cosh t)\,I + (\sinh t)\,A)\,c. \end{aligned}$$

The initial condition $c = (1, \ldots, 1)^{\mathsf{T}}$ was used to verify, with time solutions for values of n up to 50 and of m up to 100. The exact Taylor coefficients of the power series in t for the x_i were generated. The time, in seconds, to execute the program including the evaluation of the x_i for $t = 0.1\,(0.1)\,1.0,\ 1.2\,(0.2)\,2.0,\ 3.0\,(1.0)\,10.0$ is roughly approximated by $0.005\,m^{0.4}\,n^{1.4}$ on an IBM 3081 running under VM-CMS. For the idempotent case, $A^2 = A$, $f(u) = u^2 - u$, and, hence, the diagonal of M contains 0's or 1's. Bellman [16, p. 67] provides a solution for A obtained from the orthogonal matrix S_n, defined recursively by

$$S_2 = \begin{pmatrix} \cos v & -\sin v \\ \sin v & \cos v \end{pmatrix}, \qquad S_{i+1} = \begin{pmatrix} 1 & \vdots & 0 \\ \cdots & \cdots & \cdots \\ 0 & \vdots & S_i \end{pmatrix}.$$

S_n was replaced by S_n^{32} in order to reduce sparseness, i.e., $A = S_n^{32} M \left(S_n^{32}\right)^{\mathsf{T}}$. The solution for x is

$$\begin{aligned} x = \exp(At)\,c &= \left(I + At + A^2\frac{t^2}{2!} + \cdots\right)c = \left(I + A\left(t + \frac{t^2}{2!} + \cdots\right)\right)c \\ &= (I + (\exp(t) - 1)\,A)\,c. \end{aligned}$$

The solution verification in this case yielded a rough timing of $0.008\,m^{0.4}\,n^{1.6}$ seconds, under the same conditions as before.

5. APPLICATIONS

The companion matrix to the n-th order linear ordinary differential equation

$$x^{(n)} + a_{n-1}\, x^{(n-1)} + \cdots + a_0\, x^{(0)} = 0, \qquad x^{(i)}(0) = c_i, \quad (i = 0, \ldots, n-1)$$

converts it to a linear system. Let A be the companion matrix

$$A = \begin{bmatrix} 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 1 \\ -a_0 & -a_1 & \ldots & -a_n \end{bmatrix}$$

and let $x_0 = (c_0, \ldots, c_{n-1})^{\mathsf{T}}$. The program directly provides the solution to the original ODE, explicity in terms of t, without requiring the characteristic roots. This application was tested by starting with given characteristic roots, y_i, and then generating the problem.

Queueing problems described by linear stochastic equations can be explicity solved by our program, avoiding the enormous difficulty of obtaining a generating function and then evaluating its derivatives. Furthermore, one need not, only settle for the steady-state solution in those cases where the transient solution is most likely needed. For example, consider the birth-death process (i.e., the M/M/∞ queue) defined on the positive integers, i, where a and b are the infinitesimal probabilities of birth (arrival) and death (service), respectively; and where the process starts at $i = i_0$ at time $t = 0$. The differential-difference equation for this process is

$$\frac{dP_0}{dt} = -a\, P_0 + b\, P_1,$$
$$\frac{dP_i}{dt} = a\, P_{i-1} - (a + ib)\, P_i + (i+1)\, b\, P_{i+1}, \quad P_{i_0}(0) = \delta_{ii_0},$$

where $P_i(t)$ is the probability of achieving a count of i at time t. The well known solution of this problem [17] is given by

$$P_i(t) = i^{-1} \exp\left(-\frac{a}{b}\left(1 - e^{-bt}\right)\right) \sum_{k=0}^{n} \binom{n}{k} \frac{i_0!}{(i_0 - k)!} \left(\frac{a}{b}\right)^{i-k} \left(1 - e^{-bt}\right)^{i-i_0-2k} e^{-bkt}.$$

The observed execution time is approximated by $0.01\, m^{0.6}\, n^{1.6}$ seconds. The generation of the exact solution contributed substantially to the total execution time. A is an infinite matrix approximated by its first n row and columns. In this instance, A is tri-diagonal. Saaty [17] develops the solution to such queueing problems with tri-diagonal matrices by using Sylvester's spectal decomposition method. This method requires the characteristic roots.

Let $x^* = (P_0, P_1, \ldots, P_{n-1})^{\mathsf{T}}$ be the exact solution, and define the relative error by $E(t) = |x_1^* - x_1|/x_1^* + \cdots + |x_n^* - x_n|/x_n^*$, where the x_i are the components of the approximate solutions. The following table displays E vs. t, for n=20 and m=50 and where double precision floating point hardware arithmetic has been employed to evaluate the resulting expressions.

Table 1.

t	0	0.1	0.2	0.5	1.0	2.0	5.0
E	0	10^{-15}	10^{-13}	10^{-10}	10^{-7}	10^{-6}	10^{-4}

This problem is indicative of many that occur in data communication and in the analysis of operating systems. For example, the multiprogramming CPU input-output lock-out problem is similar to the one solved here, but for finite channeling. Our direct solution avoids the great complexity incurred by the usual transform methods [17, p. 110], and immediately gives a time domain solution.

6. GENERALIZING THE LINEAR CASE

The solution method for solving $dx/dt = Ax$, $x(0) = c$, has been adopted to yield a symbolic solution to more general linear cases [16], provided that pertinent terms are polynomials or can be effectively represented by polynomials. The inhomogeneous case with constant A is given by $dx/dt = Ax + b$, $x(0) = c$, with solution

$$x = \exp(At)\, c + A^{-1}\,(\exp(At) - I)\; b.$$

The only new thing here is the need to invert a constant matrix with very large precision. The symbolic matrix inverter in [18] has been used here; also, see [19–20]. The inhomogeneous case with constant A and variable b is $dx/dt = Ax + b(t)$, $x(0) = c$, and has solution

$$x = \exp(At)\, c + \int_0^t \exp\left(A\,(t-s)\right)\, b(s)\, ds.$$

In the case $b(t)$ is assumed to have polynomial components in order to facilitate symbolic integration. Finally, the variable inhomogeneous case is represented by $\dot{x} = A(t)\, x + b(t)$, $x(0) = c$, with solution

$$x = B(t)\, c + \int_0^t B(t)\,(B(s))^{-1}\, b(s)\, ds,$$

$$B(t) = \exp\left(\int_0^t A(t)\, dt\right).$$

The elements of $A(t)$ are assumed to be polynomials. The inversion of B can be symbolically performed as mentioned above [18]. The purpose of introducing these linear solutions is that the Newton-Kantorovich method reduces the nonlinear problem to sequences of linear problems.

7. NEWTON-KANTOROVICH SOLUTION OF THE NONLINEAR CASE

The author has documented the use of the Kantorovich's extension of Newton's method on Banach spaces to solve the nonlinear case [21,22]. This method produces a sequence of linear problems, which, ultimately are evaluated by the basic solution of $dx/dt = A(t, x)$, $x(0) = c$, where $A = A(t, u)$ is twice differentially continuous in u. Let $\dot{x} = dx/dt$ and write

$$P(x) = \dot{x} - A(t, x) = 0.$$

Let the initial approximation for seeding Newton's method be $x_0(t)$ and satisfy $x_0(0) = c$. Then compute the Frechet derivative operator of P at $x = x_0$,

$$P'(x_0)x = \lim\left(\frac{P(x_0 + hx) - P(x_0)}{h}\right) = \dot{x} - A_u(t, x_0)\, x.$$

Let $A'(t) = A_u(t, x_0)$, and write the Newton-Kantorovich iteration for the modified Newton's method,

$$P'(x_0)\,(x_{i+1} - x_i) = -P(x_n),$$

which yields

$$x_{i+1} = A'(t)\, x_{i+1} + b(t), \qquad x_{i+1}(0) = c,$$

where

$$b(t) = A(t, x_i) - A'(t)\, x_i.$$

$b(t)$ is purely a function of t, since x_i has been previously determined at the i-th iteration. Note that each iteration requires the solution of a linear system. The modified Newton's method has been used throughout this paper. The original Newton's method could just as well been used.

8. A NONLINEAR EXAMPLE

Consider the nonlinear system

$$\dot{x}_1 = (x_2+1)^{0.5} - 1, \qquad x_1(0) = 0,$$
$$\dot{x}_2 = (x_1+1)^2 - 3x_2 - 3, \qquad x_2(0) = 0,$$

whose solution is $x_1^* = \exp(-t) - 1$ and $x_2^* = \exp(-2t) - 1$. Let the initial approximation be $x_0 = (0,0)^\mathsf{T}$; then the first approximation is given by the solution of the resulting linear system

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}0 & 0.5\\ 2 & -3\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \left(\begin{bmatrix}0\\ -2\end{bmatrix} - \begin{bmatrix}0 & .05\\ 2 & -3\end{bmatrix}\begin{bmatrix}0\\ 0\end{bmatrix}\right), \qquad \begin{bmatrix}x_1(0)\\ x_2(0)\end{bmatrix} = \begin{bmatrix}0\\ 0\end{bmatrix}.$$

A Taylor polynomial of order m=20 has been used for this and subsequent iterations. The relative error given by $E_i = |x_1^* - x_{1i}|/|x_1^*| + |x_2^* - x_{2i}|/|x_2^*|$ for iteration i is shown in the following table.

Table 2.

t	E_1	E_2	E_3
0.0	0	0	0
0.1	$8\cdot10^{-3}$	$5\cdot10^{-4}$	$3\cdot10^{-7}$
0.2	$2\cdot10^{-2}$	$3\cdot10^{-3}$	$3\cdot10^{-6}$
0.5	$9\cdot10^{-2}$	$9\cdot10^{-3}$	$8\cdot10^{-5}$
1.0	$1\cdot10^{-1}$	$4\cdot10^{-2}$	$6\cdot10^{-5}$
2.0	$3\cdot10^{-2}$	$7\cdot10^{-4}$	$3\cdot10^{-5}$
5.0	$1\cdot10^{0}$	$9\cdot10^{-2}$	$5\cdot10^{-4}$

It is interesting to note the similarity of this solution to a solution by the method of Green's functions, specifically, the role of $A'(t,u)$ as a Green's function [23].

9. THE CONSTANT JACOBIAN MATRIX CASE

The nonlinear problem, $\dot{x} = A(t,x)$, $x(0) = c$, can be transformed by replacing x by $x - c$, to obtain a new problem with $c = 0$. Hence, if the initial Newton's approximation is taken to be $x_0 = c = 0$, the Jacobian matrix, A, is often constant. In this case, the differential equation for x_{i+1} can be readily solved for i=0 to yield

$$x_1 = \exp(A't)\int_0^t \exp(-A's)\,A(s,0)\,ds.$$

If it happens that x_1 is sufficiently accurate over some range $[0,h)$, then rather than iterate on x_n starting from x_1, it would be simpler to increment $t = 0, h, 2h, \ldots$, in order to obtain piecewise solutions from x_1 in the intervals $[0,h)$, $[h,2h)$, etc. Let y denote the following approximation to $x_1(t)$ in the range $[0,h)$,

$$y_1 = (I + A't)\int_0^t (I - A's)\,A\left(\frac{h}{2},0\right)ds$$
$$= tA(\tfrac{h}{2},0)\,(I + A't)\,(I - 0.5A't),$$

where the first two terms of the MacLaurin expansion of the exponential function have been used, and $A(t,0)$ has been approximated by its half range value. Let y_i be this approximate solution on the i-th interval, $[(i-1)\,h, ih)$, and c_i the initial conditions for this interval, then

$$c_i = y_{i-1}\,((i-1)\,h)$$

and

$$y_i = (I + A't)\, c_i + (I + A't) \int_{(i-1)h}^{ih} (I - A's)\, (A((i - 0.5)\, h, c_i) - A'c_i)\, ds$$
$$= (I + A't)\, c_i',$$

where

$$c_i' = c_i + h\, (I - (i - 0.5)\, hA')\, (A((i - 0.5)h, c_i) - A'c_i)\,.$$

This simple procedure, of course, yields the Adams point-slope method. The y_i may be further refined by performing Newton iterations on

$$x_0 = \sum_{i=1}^{n} \text{step}\,((i-1)\, h, y_i, ih)\,,$$

where the step function takes the values of its middle argument in the interval $[(i-1)\, h, ih)$. This piecewise solution of the previous nonlinear problem for $h = 0.2$ yields accuracy roughly comparable to the results obtained from two Newton iterations. This method would yield much greater accuracy if the series expansion of $\exp(At)$ were to retain more terms. In fact, the basic solution provides the exact evaluation of the expression for x_1.

References

1. C. Underhill and A. Wragg, Convergence properties of Padé approximation to $\exp(z)$ and their derivatives, *J. Inst. Maths. Applications* **11**, 361–367 (1973).
2. J.I. Siemieniuch, Properties of certain rational approximations to $\exp(-z)$, *BIT* **16**, 172–191 (1976).
3. E.B. Saff, R.S. Varga and W. Ni, Geometric convergence of rational approximations to $\exp(-z)$ in infinite sectors, *Numerische Mathematik* **26**, 211–255 (1976).
4. S.P. Norsett, C-polynomials for rational approximations to the exponential function, *Numerische Mathematik* **25**, 39–56 (1975).
5. A.D. Ziebur, On determining the structure of A by analyzing $\exp(At)$, *SIAM Review* **12**, 96–102 (1970).
6. T.M. Apostol, Some explicit formulas for the exponential matrix $\exp(At)$, *Am. Math. Monthly* **76**, 288–292 (1969).
7. A. Wragg and C. Davies, *J. Maths., Inst. Applications* **11**, 369–375 (1973).
8. R.C. Ward, Numerical computation of the matrix exponential with accuracy estimate, Oakridge National Laboratory Report, (Nov. 1975).
9. W.J. Cody, G. Meinardus and R.S. Varga, Chebyshev rational approximations to $\exp(-x)$ in $[0, \infty)$ and applications to heat-conduction problems, *J. Approximation Theory* **2**, 50–65 (1969).
10. C.B. Moles and C.F. Van Loan, Nineteen dubious ways to compute the exponential of a matrix, *Siam Review* **20**, 801–836 (1978).
11. FORMAC, Software order no. 360D-03.3013, Share Program Library Agency, Research Triangle Park, NC, (1975).
12. J.N. Hanson and F. Wolff, Variable very high precision arithmetic, (to appear).
13. L.V. Kantorovich and G.P. Akilov, *Functional Analysis in Normed Spaces*, Pergamon Press, NY, (1964).
14. J. Hanson, A. Benander and B. Benander, The computer generated symbolic solution of a system of linear first order differential equations, *Computers and Mathematics with Applications* **19** (7), 7–12 (December 1990).
15. D.K. Faddeeva and V.N. Faddeev, *Computational Methods of Linear Algebra*, W.H. Freeman and Co., San Francisco, (1962).
16. F.R. Gantmacher, *Theory of Matrices*, Chelsea, NY, (1959).
17. R.K. Bellman, *Introduction to Matrix Analysis*, McGraw-Hill, NY, (1960).
18. T.L. Saaty, *Elements of Queueing Theory with Applications*, Dover Publications, NY, (1961).
19. J.N. Hanson, Computer aided symbolic solution of multipoint boundary value problems occurring in Physics and Engineering, *Computer Methods in Applied Mechanics and Engineering* **25**, 149–178 (1981).
20. S. Cabay, Exact solution of linear equation, Presented at the *Proc. Second Symp. on Symbolic and Algebraic Manipulation*, ACM, Los Angeles, pp. 392–398, (March 1971).
21. J.P. Lipson, Symbolic methods for the solution of linear equations with applications to flowgraphs, *Proc. 1968 Summer Inst. of Symbolic Computation*, IBM Programming Lab. Report FSC 69-0312, (1969).
22. J.N. Hanson, Experiments with equation solutions by functional analysis algorithms and formula manipulation, *J. Computational Physics* **9**, 25–52 (1972).
23. J.N. Hanson, Functional analysis, formula manipulation, and satellite geodesy, *J. Geophysical Research* **78**, 3260–3270 (1973).